Automatització i Ciberseguretat a Casa

NoMoreDoc

Joaquin Bosch Corta
1455228
Curs 2019-2020

Resum: Quan s'intenta atacar un sistema cal conèixer el seu funcionament a la perfecció. Cal realitzar un estudi complet de la víctima per optimitzar cada una de les proves de penetració. És per això que la fase d'enumeració és la més important. El pentester ha de dedicar gran part del seu temps en reconèixer les característiques que descriuen al seu objectiu i convertir-les en la base del seu atac. Aquestes tasques d'estudi normalment es combinen amb tasques de documentació per enregistrar de manera permanent tota la informació descoberta. D'aquesta manera quan passem a dissenyar l'atac evitem perdre detalls importants per assolir l'èxit. El problema és que realitzar una bona documentació requereix dedicar cert temps el qual es podria aprofitar per millorar l'estudi de la víctima o finalitzar-lo abans. Aquest document presenta una eina anomenada NoMoreDoc dissenyada per automatitzar l'enregistrament de la informació recollida pel pentester de manera paral·lela i simple. A més aquest sistema es basa en mòduls fàcilment modificables i ampliables per tal d'oferir la màxima personalització i optimització a l'usuari final. Per últim, un punt destacable d'aquesta eina és la seva integració amb Alexa i la seva possibilitat d'explotar totes les funcionalitats que poden donar les interaccions per veu.

Paraules clau: Documentació, Automatització, Enumeració, Ciberseguretat, Pentesting, Red Team, Python, Alexa.

Abstract: When trying to attack a system, you have to know how it works perfectly. A complete study of the victim must be carried out to optimize each penetration tests. That is why the enumeration phase is the most important. The pentester must spend much of his time recognizing the characteristics that describe his target and making them the basis of his attack. These study is normally combined with documentation tasks to permanently record all the information discovered. In this way when we go on to design the attack we avoid losing important details to achieve success. The problem is that making good documentation requires spending some time that could be used to improve the study of the victim or finish it earlier. This document presents a tool called NoMoreDoc designed to automate the documentation of the information collected by the pentester in a parallel and simple way. Furthermore, this system is based on easily modifiable and expandable modules to offer maximum customization and optimization to the end user. A highlight of this tool is its integration with Alexa and its possibility of exploiting all the functionalities that voice interactions can give.

Paraules clau: Documentation, Automation, Enumeration, Cybersecurity, Pentesting, Red Team, Python, Alexa.

———————— ◆ ————————

1. INTRODUCCIÓ

La principal motivació al món de la ciberseguretat ofensiva és el fet d'aconseguir informació que inicialment es desconeixia i no es tenia accés per així solucionar aquesta debilitat i evitar que un atacant maliciós pugui realitzar la mateixa acció. S'ha pogut comprovar que la tasca més difícil en un test d'intrusió (en anglès Penetration Test[1] i des d'ara pentest) per als professionals (des d'ara pentesters) és la documentació; i no pas per la seva complexitat, sinó pel fet d'haver de deixar a un costat l'atac i dedicar-se a explicar el que s'està fent i aconseguint. Aquesta és la principal motivació d'aquest treball: desenvolupar una eina que permeti al pentester divertir-se i atacar mentre, de manera automàtica, es van enregistrant els resultats obtinguts. Clar està que no podrem crear així un perfecte informe tècnic, i molt menys un d'executiu, però sí

que estalviem les llargues pauses de documentació entre atacs substituint-les per la recol·lecció dels resultats i mostrant-los de manera gràfica i útil. D'aquesta manera només caldrà consultar la web al final per realitzar els informes.

Inicialment es tenia clar que es volia desenvolupar una eina automàtica dedicada al món de la seguretat[2]. La primera idea era orientar-la per al Blue Team (Equip dedicat a la defensa de la infraestructura), amb funcionalitats de monitoratge, alerta i fins i tot resposta. No va ser fins poques setmanes abans de tenir la primera reunió amb el meu tutor que es va decidir orientar-la al món contrari, el Red Team (Equip dedicat a simular possibles enemics mitjançant atacs a la infraestructura) i permetre que aquest pugui modular-la afegint els scripts que es vulgui per tal de realitzar els tests d'intrusió centrant-se només en l'estudi de l'objectiu i gaudir de l'atac. A més es pretén donar al pentester la possibilitat d'executar ordres bàsiques per veu amb l'ajuda d'Alexa[3] i alhora oferir a l'usuari final la possibilitat de millorar i ampliar aquesta funcionalitat.

Aquesta eina s'ha desenvolupat per entorns Linux per la seva facilitat de manipulació i perquè actualment es disposa més coneixement que per altres entorns. El llenguatge de programació principal és Python per la seva portabilitat i potència amb les llibreries. La base de dades s'aixeca amb PostgreSQL i la web amb Python. La principal idea és potenciar la portabilitat i modulabilitat de l'eina i s'evitaran les solucions complexes.

A continuació es mostren els objectius principals que es volia assolir amb aquesta eina i tot el disseny i procediment que s'ha seguit per fer-la real. També es dóna una visió general del mon on es situa aquesta eina per demostrar la seva raó de ser i el seu potencial.

2. OBJECTIUS

A continuació s'exposen els diferents objectius del treball. Cal esmentar que s'han endreçat per ordre de la seva prioritat de manera descendent, és a dir, els primers objectius tenen una importància superiors respecte als darrers.

1. Creació de la Base de Dades que desarà els resultats dels scripts:
S'ha creat una base de dades per desar els resultats obtinguts de les proves i per fer possible la comunicació entre l'entorn dels scripts i la web. Les tasques realitzades són:

1. Disseny de la Base de Dades.
2. Creació de la Base de Dades.
3. Creació del script que desarà resultats a la base de dades.

2. Creació de l'entorn que processarà les ordres de l'usuari i executarà els scripts:
S'ha creat un entorn que assigna cada script del sistema amb una ordre única. D'aquesta manera quan l'usuari executi una ordre el sistema executarà el script adient i, en el cas que l'usuari hagi afegit paràmetres, s'inclouran a l'execució d'aquest. Les tasques han estat les següents:

4. Disseny de l'entorn.
5. Implementació bàsica de l'entorn.
6. Creació del primer script (combinat amb nmap).
7. Creació del script que permetrà a l'usuari afegir els seus propis scripts.

3. Creació de la interfície web que mostri el contingut de la base de dades:
S'ha programat una web per representar de manera visual i intuïtiva el contingut de la base de dades per tal de facilitar la recol·lecció dels resultats al pentester. S'han detallat les següents tasques:

8. Disseny de la interficie.
9. Disseny de les consultes de la Base de Dades.
10. Desenvolupament de les consultes.
11. Desenvolupament de la interfície.

4. Creació de nous Scripts i funcionalitats:
S'han desenvolupat algunes funcionalitats per millorar l'eina i demostra la seva facilitat de personalització. No es pretén desenvolupar funcionalitats altament complexes, simplement es vol demostrar la gran potencia que proporciona aquesta eina.

5. Integració amb alexa de tal manera que l'usuari pugui executar ordres per veu:
Per tal de facilitar l'ús es vol permetre que el pentester executi ordres simples per veu. Aquestes ordres seran les que no requereixen arguments o que la seva crida no és altament complexa

Envers la situació actual donada pel Covid-19 s'ha tingut la sort que el treball no s'ha vist afectat. Pel que fa als objectius i a la planificació segueix tot igual. L'única diferència és s'hadisposat de més temps per dedicar a aquest projecte.

3. ESTAT DE L'ART

Actualment es desconeix l'existència d'una eina comercial tan bàsica i modulable com la que es pretén desenvolupar. La idea va sorgir perquè es va treballar amb una empresa que havia

creat per al seu propi ús un entorn d'enumeració amb funcionalitats similars. D'aquesta manera es va decidir implementar-la permetent que qualsevol usuari amb els coneixements necessaris la pogués personalitzar afegint els seus propis scripts.

Pel que fa als llenguatges de programació, Python és conegut per tenir infinites possibilitats amb les diferents llibreries de les quals disposa. D'aquesta manera simplement caldrà trobar la documentació necessària i aplicar-la puntualment al treball.

La Base de Dades s'ha aixecat amb l'ajuda del sistema de gestió anomenat PostgreSQL per la seva facilitat d'ús i pel coneixement previ del qual es disposa. A més s'ha trobat una llibreria de Python dedicada a l'automatització de les diferents interaccions amb les bases de dades d'aquest sistema.

Pel que fa a la pàgina web, s'ha desenvolupat un servidor amb Python, per així reutilitzar el codi de per gestionar la base de dades, i se li dona forma amb HTML5 i CSS. Això és degut al fet que al llarg de la carrera s'han realitzat pràctiques de desenvolupament de webs amb objectius similars.

Per últim, se li ha volgut afegir la funcionalitat de poder executar instruccions per veu. Això es realitza amb l'ajuda d'Alexa. Es disposa d'un compte de desenvolupador d'Amazon (totalment gratuïta i accessible per tothom) amb la qual es podrà programar les anomenades "Skills". Aquestes funcions també seran programades amb Python. Amazon disposa d'una àmplia documentació formada per documents de text i vídeos perfectament detallats[4].

4. METODOLOGIA

A l'haver dividit la feina en tasques tan específiques s'ha decidit utilitzar la metodologia SCRUM[5]. Els esprints eren d'una setmana cada un i es realitzava la revisió al cap de setmana com a molt tard. S'han estudiat altres metodologies, però com s'iterarà sempre sobre la feina realitzada anteriorment i les tasques a realitzades són molt específiques es va arribar a la conclusió que la metodologia Scrum és la més adequada per aquest projecte. A més es té molt present que la possibilitat que surtin imprevistos al llarg del desenvolupament d'aquest projecte és bastant alta. Tot i que els objectius i tasques estaven definits no es disposava d'un perfecte coneixement de les capacitats de cada una de les parts. Aquesta metodologia està principalment dissenyada per suportar tots els canvis que un client pot anar demanant al llarg del desenvolupament. A més, al realitzar aquest treball amb una empresa, aquesta requeria monitorar constantment el progrés. I quan ha sigut necessari modificar algun dels objectius s'ha tingut prou coneixement del projecte per implementar-ho de la manera més eficient i senzilla.

La realització de cada tasca es dividia en quatre fases molt diferents. La primera era la de l'estudi de l'objectiu i la recollida d'informació necessària per al seu desenvolupament. A continuació es realitzava el corresponent disseny de desenvolupament i test. Seguidament es passava a la fase del desenvolupament de la tasca, on es realitza la feina necessària per assolir els objectius d'aquesta. I finalment la fase de test, on es validava el desenvolupament realitzat amb les proves especificades anteriorment.

5. PLANIFICACIÓ

Com es pot comprovar el treball es basa principalment en el desenvolupament de tres infraestructures diferents: un entorn amb scripts a executar per estudiar els objectes d'estudi del pentest, una base de dades amb els resultats d'aquests scripts, i un portal web que mostri de manera visual i gràfica aquests resultats. A més es pretén oferir l'opció d'executar ordres simples per veu amb ajuda d'Alexa. D'aquesta manera resulta més fàcil fraccionar la feina a realitzar. Es va realitzar un estudi dels objectius d'aquest treball i per aconseguir l'èxit es va decidir dividir la feina en les següents activitats:

Base de Dades:

1. Disseny de la Base de Dades.
2. Creació de la Base de Dades.
3. Creació del script que desarà resultats a la base de dades.

Entorn:

4. Disseny de l'entorn.
5. Implementació bàsica de l'entorn.
6. Creació del primer script (combinat amb nmap).
7. Creació del script que permetrà a l'usuari afegir els seus propis scripts.

Web:

8. Disseny de la interfície web.
9. Disseny de les consultes a la BD.
10. Creació de les consultes a la BD.
11. Creació de la interfície web.

Scripts i funcionalitats:

12. Disseny de nous scripts.
13. Implementació de nous scripts.

Alexa:

14. Integració amb alexa per executar comandes per veu.

6. DESENVOLUPAMENT

Com ja s'ha comentat anteriorment, l'objectiu principal és desenvolupar les tres parts principals del treball. Com es veu a la següent figura, la finalitat és aconseguir crear un entorn que desi la informació trobada a la base de dades de manera automàtica, aquest entorn també hauria de poder consultar el contingut d'aquesta base de dades en el cas que sigui necessari. Per altra banda s'ha de disposar d'una web accessible per l'usuari que mostri de manera visual i esquemàtica el contingut de la nostra base de dades.

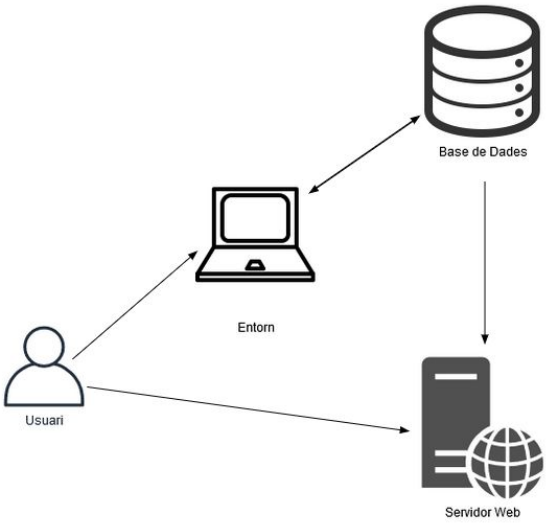

Figura 1: diagrama amb els components de l'eina

Es vol millorar la funcionalitat de l'entorn implementant un controlador anomenat Job Controller. Aquest serà un agent que desarà totes les ordres que executa l'usuari i les anirà executant de manera paral·lela respectant sempre el nombre màxim de threads indicat.

Un cop s'hagi desenvolupat aquest sistema es passarà a crear scripts per ampliar les funcionalitats de l'entorn i es desenvoluparà la integració amb Alexa.

A continuació s'exposen les diferents parts a desenvolupar dividides en 4 apartats. Primer es mostra la idea i objectius de cada component a l'anàlisi. Seguidament es s'explicarà d'una manera més técnica el seu disseny. A continuació es fa un resum de l'implementació i de les diferents dificultats que han anat sorgint durant tot el procés. I per últim s'exposen les diferents proves que s'han realitzat i els resultats obtinguts per tal de demostrar la perfecte funcionalitat.

6.1. BASE DE DADES

Anàlisi:

S'ha aixecat una base de dades per desar la informació obtinguda a la fase d'enumeració dels pentests realitzats per l'usuari. Es vol aconseguir una estructura de dades modular i resistent a possibles canvis posteriors.

Bàsicament el tipus d'informació que es vol desar són Pentests realitzats a diferents llocs (des de ara Sites) on s'identifiquen diferents Hosts amb els seus corresponents ports oberts i serveis que proporcionen.

Disseny:

Amb l'anàlisi anterior s'han extret 6 classes necessàries per representar les dades:

- Pentests
- Sites
- Ports
- Services (Serveis)
- Credentials (Credencials)

Aquesta darrera ens serà útil per desar paraules d'accés o usuaris que ens puguin servir per accedir a serveis proporcionats.

A part s'ha creat la classe Credentials_Ports per facilitar la relació entre les taules Credentials i Ports permetent així disposar de n relacions entre aquestes. A més s'ha decidit usar una estructura d'herència amb la classe Services per facilitar la creació de diferents tipus de serveis mantenint una estructura base.

L'estructura de la base de dades és la següent:

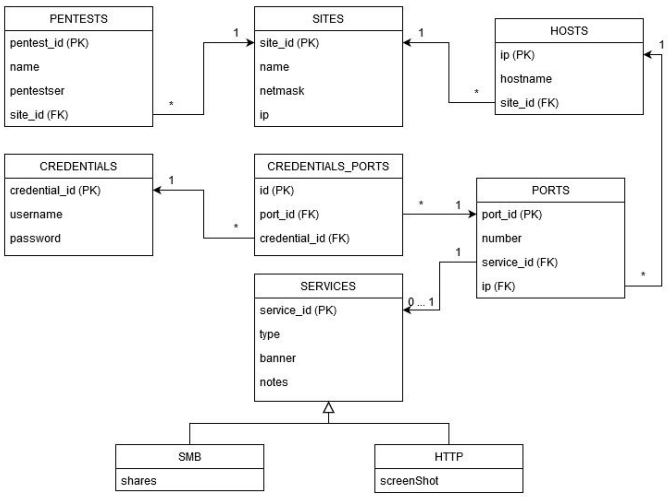

Figura 2: estructura de la base de dades

Implementació:
Finalment és una base de dades basada en PostgreSQL, inicialment es volia aixecar amb MySQL, però aquest no permet relacions d'herència com les que es requereixen per aquest projecte. De la mateixa manera es disposa d'una llibreria de Python[6] per operar i automatitzar totes les interaccions amb les dades desades.

Gràcies a la llibreria esmentada s'ha creat una llibreria pròpia amb totes les funcions necessàries per interactuar amb les diferents taules que formen la base de dades d'aquest projecte.

Test:
Les proves realitzades amb la base de dades han estat bastant simples. S'han guardat diferents classes de prova i s'ha comprovat que existís una correcta relació entre elles. A més s'ha pogut comprovar que els resultats obtinguts amb les consultes eren idèntics als esperats.

6.2. ENTORN

Anàlisi:
S'ha creat una consola interactiva que permet a l'usuari executar ordres concretes, simples i fàcils de recordar. És important que l'usuari tingui total control sobre el qual s'està executant i se li mostri tota la informació referent a aquestes execucions.

A més l'usuari és perfectament capaç d'ampliar les funcionalitats de l'entorn sense necessitar accedir al codi font de l'entorn. Es disposar de diferents ordres que li permeten gestionar aquestes funcionalitats personalitzades.

Aquest entorn és totalment capaç de desar de manera automàtica i independent tota la informació descoberta per l'usuari.

Disseny:
L'entorn s'ha programat amb Python per la seva senzillesa i la seva excel·lent potabilitat. D'aquesta manera es s'aprofita la llibreria anteriorment programada per operar amb la base de dades. Bàsicament serà un programa que espera l'ordre introduïda per l'usuari, comprova si és capaç de processar-la i, en cas afirmatiu, executa la funcionalitat pertinent.

El programa deixarà de demanar ordres a l'usuari quan aquests introdueixi les ordres *exit* o *quit*.

Implementació:
A l'entorn se li ha donat el nom de *NoMoreDoc* (de l'anglès No More Documentation), expressant així la finalitat d'estalviar la documentació al pentester.

Quan iniciem l'entorn se'ns demanarà la següent informació:

```
root@kali:~/Hacking/tfg# python noMoreDoc.py
database:
userDB:
passwordDB:
host:
port:
```

Figura 3: Execució de noMoreDoc.py

- **database:** Nom de la base de dades on es consultarà i desarà la informació.
- **userDB:** Usuari de la base de dades.
- **passwordDB:** Contrasenya de l'usuari introduit anteriorment.
- **host:** Direcció del host on es troba la base de dades. En aquest cas com la base de dades es troba en local introduirem 127.0.0.1 o localhost.
- **port:** Port del host introduit anteriorment que escolta i proporciona el servei de la base de dades. En aquest cas com es tracta d'una base de dades basada en PostgreSQL i no s'ha modificat la seva configuració introduirem el port 5432.

Cal dir que és necessari que la base de dades estigui activa en el moment que iniciem l'entorn.

Un cop introduïdes totes les dades anteriors s'obrirà una consola com la de la figura següent. A l'exemple s'ha executat la instrucció *show commands* per mostrar el llistat d'ordres que processa aquesta eina actualment i a continuació s'explica la funcionalitat de cada una d'elles.

```
xino@nomoredoc> show commands
Job Controller:
        jobctrl status
        jobctrl stop
        jobctrl start

Commands:
        add command
        remove
        show commands

Alexa:
        alexa start

Web:
        web start

User Commands:
        nmap
                [python nmap.py]

DataBase:
        show pentests
        reset db
        new pentest
        load pentest
        exit pentest
        current pentest
        show hosts
        delete pentest
```

Figura 4: resultat d'executar "show commands" a l'entorn

Interacció amb el Job Controller: Ordres destinades al control del Job Controller. Amb aquesta funcionalitat podrem executar ordres de manera paralela i optimitzar el temps.

- **jobctrl status:** Mostra l'estat del Job Controller. En el cas que aquest es trobi actiu mostrarà el nombre màxim de Threads inicialitzat per l'usuari, el nombre de jobs que esperen per ser executats i el nombre de jobs actius.
- **jobctrl stop:** Atura el Job Controller. En el cas que hi hagin jobs en alguna de les cues es pregunta a l'usuari si vol esperar a que aquests acabin. En el cas que l'usuari respongui de manera afirmativa s'aturarà el Job Controller de manera automàtica quan les dues cues estiguin buides.
- **jobctrl start:** Inicia el Job Controller. Quan l'usuari executa aquesta ordre se li preguntrà pel nombre màxim de jobs que vol executar de manera paral·lela. En el cas que l'usuari no introdueixi el nombre màxim no existirà restricció.

Gestió de funcionalitats personalitzades: Ordres destinades a la gestió de les funcionalitats que l'usuari final pot incorporar, modificar i eliminar.

- **show commands:** Mostra aquest llistat.
- **add command:** Permet a l'usuari afegir una ordre a aquest llistat. Se li demanarà:
  - El fitxer amb el codi
  - El llenguatge de programació
  - El nom amb el qual es vol cridar a la nova funcionalitat

  D'aquesta manera quan l'usuari executi la crida de la funció el programa sabrà com tractar-ho. Un exemple seria la següent instrucció.
- **nmap:** Aquest cas es tracta d'una instrucció que podria haver estat afegida per l'usuari final. La seva funcionalitat bé donada per l'script nmap.py. Aquest s'encarrega de cridar l'eina nmap i desar els resultats a la base de dades de manera totalment transparent per l'usuari. Quan l'usuari executa la instrucció nmap l'entron executa la ordre python nmap.py. En el cas que l'usuari vulgui complementar aquesta crida amb arguments les crides serien nmap <args> per part de l'usuari i python nmap.py <args> per part de l'entorn.
- **remove:** Permet a l'usuari eliminar una de les ordres introduïdes. No permet esborrar funcionalitats legítimes.

Integració amb Alexa: Aquestes ordres permeten escoltar a Alexa per executar funcionalitats per veu.

- **alexa start:** Inicia un servidor al port 1234 on espera ordres enviades per HTTP.

Web: Ordres dedicades al funcionament del servidor web.

- **start web:** Aquesta instrucció aixeca el servidor web que servirà per consultar la informació existent a la base de dades de manera simple i gràfica. Actualment no es troba del tot finalitzada, ja que es vol aconseguir que el procés del servidor s'executi en segon pla per tal de poder seguir usant l'entorn.

Interacció amb la Base de Dades: Funcionalitats necessàries per operar amb el contingut de la base de dades.

- **show pentests:** Mostra els pentests que es troben a la base de dades.
- **show hosts:** Mostra els hosts que pertanyen al Site on s'està realitzant el pentest. En el cas que no s'hagi carregat un pentest mostra un missatge d'error.
- **load pentest:** Carrega el pentest referent a l'identificador proporcionat.
- **reset db:** Esborra tot el contingut de la base de dades. Elimina dades, relacions i taules i torna a crea-la de nou. Aquesta instrucció és molt útil quan fem qualsevol modificació en el fitxer de configuració de la base de dades.
- **new pentest:** Permet crear una nova instància de pentest. Un cop creat es carrega com a pentest actual.
- **current pentest:** Mostra la informació del pentest que s'ha carregat i on es desa tota la informació trobada. En el cas que no s'hagi carregat cap pentest mostra un missatge d'error
- **delete pentest:** Elimina de la base de dades el pentest al que fa referència l'identificador proporcionat.

Actualment l'estat de l'entorn es considera finalitzat. Però cal acabar de millorar la seva interacció amb el servidor web, i es vol ampliar les seves funcionalitats amb scripts com el cas de la instrucció nmap.

Test:

Per validar aquest mòdul s'ha simulat la interacció d'un usuari final real. S'han executat cada una de les ordres disponibles i s'ha comprovat que els resultats coincideixin amb els esperats.

També s'ha validat la perfecta funcionalitat de l'ordre personalitzada *nmap*. Quan aquesta s'executa es comporta i mostra el resultat com si l'usuari hagués executat l'eina legítima de manera directa, i a més desa aquest resultat a la base de dades.

6.3. WEB

Anàlisi:

L'objectiu és aixecar un servidor web que mostri de manera gràfica i simple el contingut de la base de dades. La seva única

funcionalitat serà la de consulta, no s'hauria de poder modificar ni eliminar la informació desada a la base de dades.

Disseny:
Es programarà un servidor web usant sockets amb el llenguatge de programació Python, d'aquesta manera es podrà reutilitzar la llibreria programada anteriorment per operar amb la base de dades.

La web s'aixecarà per ordre de l'usuari a través de l'entorn abans descrit i s'aturarà de la mateixa manera.

Implementació:
S'ha programat un servidor que obre el port 9876 i proporciona els documents demanats pel client. En el moment que l'usuari executa la instrucció *start web* a l'entorn se li demanarà que introdueixi unes credencials. Aquestes serviran per fer login a la web i accedir al seu contingut. D'aquesta manera es pretén que l'usuari que accedeix a aquesta web sigui el mateix que l'ha aixecat. A més, com les credencials s'indiquen en el moment d'arrencar el servidor evitem desar-les en fitxers de configuració i permet que cada vegada que s'inicia la web siguin diferents. Aquesta informació es desa a una variable i quan s'acaba la seva execució s'elimina.

Aquest servidor usa la llibreria de sockets de Python[7] per rebre i enviar les dades al client. Com el client utilitza el protocol http ,ja que accedeix des de un navegador, s'ha programat de manera manual el processament de les peticions i la construcció de les respostes. A més la web ha estat programada per tal que el client demani els fitxers pels seus identificadors, d'aquesta manera obliguem que el servidor només processi els identificadors desitjats i eliminem la consulta de documents sensibles. A més la majoria dels documents que pot demanar el client no existeixen al servidor, per tal d'optimitzar l'espai el servidor construeix els documents i els envia quan el client els demana.

Actualment la web es troba finalitzada pel que fa a la seva perfecta funcionalitat. Finalment s'ha optat per realitzar una interfície d'usuari simple i bàsica com la que es mostra a la següent figura. De la mateixa manera que a la resta del projecte, s'ha preferit donar més importància i valor a la funcionalitat que a l'apariencia.

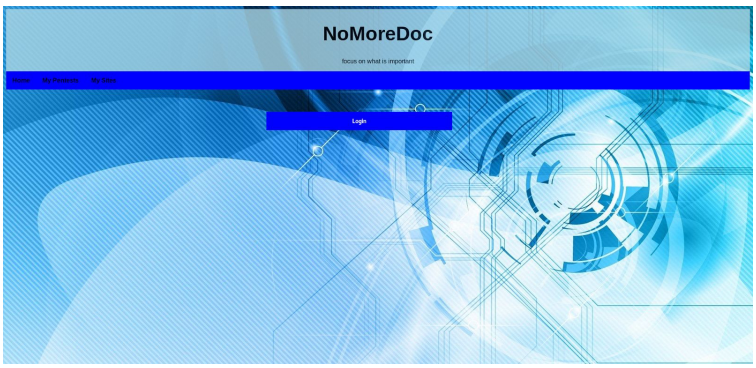

Figura 5: Pantalla de login de la web

Test:
Les proves realitzades demostren que la web funciona perfectament. Aquesta és capaç de mostrar tot el contingut de la Base de Dades de manera simple i óptima. A més gràcies a la seva implementació i al control de les peticions que proporciona aquesta, podem afirmar que la web és capaç de protegir-se envers usuaris malintencionats.

6.4. SCRIPTS I ALTRES FUNCIONALITATS

Actualment es disposa de diferents scripts perfectament funcionals on només cal enllestir el procés de desar els resultats a la base de dades. Els scripts dels quals es disposa actualment són:

- **nmap.py [Python]:** Va ser el primer script desenvolupat per garantir i demostrar el funcionament del sistema i la perfecta comunicació entre cada una de les seves parts. Aquest script simplement es basa en la crida d'una de les eines d'enumeració més conegudes anomenada Nmap. Un cop finalitza la seva execució es crea un fitxer amb el resultat. Seguidament s'extreu i es selecciona la informació i finalment, amb l'ajuda de la llibreria propria programada anteriorment, es desen els resultats a la base de dades de manera estructurada i ordenada.

- **netscan.py [Python]:** Aquest script mostra tots els hosts actius per ping o aquells que tenen el port indicat obert. En el cas que el port sigui el 445 (SMB) i s'hagi indicat que es vol accedir a aquest mostrarà també els arxius compartits. També disposa de la possibilitat d'introduir credencials per intentar accedir a aquests arxius. Actualment aquesta fuincionalitat es troba perfectament integrada amb l'eina.

6.5. INTEGRACIÓ AMB ALEXA

Anàlisi:
Es vol aconseguir que l'usuari executi ordres simples per veu mitjançant Alexa. Per fer-ho s'ha pensat crear un servidor que

escolti peticions mitjançant identificadors per a cada ordre i les executi. S'ha decidit que el resultat únicament es desarà a la base de dades i retornarà una com a resposta un missatge afirmatiu si l'ordre s'ha executat correctament o amb un de negatiu en el cas d'error, ja que la majoria dels resultats de les ordres no estan preparats per ser retornats a Alexa pel su format o pel temps que es triga a obtenir-los.

Disseny:
S'aixeca un servidor HTTP com el de la web programada anteriorment que escoltarà les peticions del servidor d'Alexa. D'aquesta manera es permetrà executar ordres de manera remota desde qualsevol navegador o fins i tot permetre l'automatització d'aquestes peticions. Les Skills que es programaran es basaran en fer una petició HTTP al nostre servidor amb l'identificador de l'ordre demanada per l'usuari.

Implementació:
Aprofitant el codi del servidor web s'ha creat un de nou. S'ha adaptat el codi per tal que executi les ordres personalitzades per l'usuari i es comuniqui amb alexa. Aquest servidor es basa en el protocol HTTP i escolta al port 1234.

Les Skills programades es basen en la frase "Alexa ejecuta <ordre>". Quan l'usuari diu aquesta frase Alexa fa una petició HTTP al nostre servidor amb l'identificador de l'ordre corresponent. Aquestes skills han estat programades en Python amb l'ajuda de les eines de desenvolupador que proporciona Amazon de manera gratuita. El codi escrit per aquesta skill és bastant simple gràcies a les llibreries i funcions proporcionades per la plataforma. Simplement cal tractar les ordres d'entrada que venen en format de String i crear una resposta amb el mateix format per tal que sigui llegida per Alexa. Pel que fa al comportament de cada skill simplement s'utilitza la llibreria requests[8] de python per fer la petició al nostre servidor i recollir el resultat. A continuació es mostra el codi de la funció que interactua amb la funcionalitat de nmap. Com no s'envien paràmetres simplement es retornaraia un missatge de 'Done' si la petició ha arribat al servidor i l'identificador correspon a una de les ordres disponibles.

```
class nmapIntentHandler(AbstractRequestHandler):
    """Handler for nmap Intent."""
    def can_handle(self, handler_input):
        # type: (HandlerInput) -> bool
        return ask_utils.is_intent_name("nmap")(handler_input)

    def handle(self, handler_input):
        # type: (HandlerInput) -> Response
        try:
            res=requests.get('https://<ip del servidor>:1234/nmap').content
        except Exception as e:
            res=str(e)
        speak_output=srt(res)

        return (
            handler_input.response_builder
                .speak(speak_output)
                # .ask("add a reprompt if you want to keep the session open
                .response
        )
```

Figura 6: Codi de la Skill d'Alexa que executa l'ordre "nmap" a l'entorn

Aquesta funcionalitat s'activa quan l'usuari executa l'ordre *alexa start* a l'entorn.

Test:
El nostre servidor funciona perfectament, en aquest moment podem executar ordres de manera remota desde qualsevol navegador.

De la mateixa manera, actualment es pot realitzar qualsevol petició des de Alexa. De moment el servidor es troba en una xarxa local i proporciona el seu servei mitjançant el protocol NAT. Alexa realitza les peticions al router de la xarxa i aquest les redirigeix al nostre servidor.

7. PRESENTACIÓ I DISCUSSIÓ DELS RESULTATS

Actualment es disposa d'una éina altament funcional i preparada per a que qualsevol usuari amb els coneixements necessaris programi les seves funcionalitats personalitzades i les inclogui a l'entorn.

Pel que fa als resultats obtinguts les sensacions són altament satisfactòries, ja que fa uns mesos es va començar el projecte amb una carpeta buida i ara es disposa de dos servidors, una base de dades i un entorn perfectament funcionals. Aquest treball ha servit per posar en pràctica els coneixements teòrics obtinguts sobre bases de dades, servidors web i el protocol http a l'igual que millorar la programació amb Python i explotar més les seves capacitats.

Finalment també es vol comentar l'èxit envers la integració d'Alexa amb NoMoreDoc. Tot i que la solució de les dificultats trobades anteriorment es basava en la correcció i reestructuració del codi ja existent, s'ha aconseguit un nou camí disponible per ampliar i millorar la personalització de l'eina. Alexa permet executar ordres sense haver d'accedir de manera física a NoMoreDoc. La complexitat d'aquestes ordres dependrà de manera directe de les funcionalitats de l'usuari final. Amb aquest treball s'ha aconseguit donar l'opció d'explotar aquesta via de la manera més fàcil i simple per al pentester.

Aquest projecte ha servit per potenciar les capacitats de l'autor de treballar de manera efectiva en entorns desconeguts assolint l'èxit amb els resultats obtinguts. Ha obligat a portar grans estudis previs i constants al llarg del projecte per desenvolupar el sistema correctament i potenciant les possibilitats que pot oferir cada una de les parts desenvolupades.

8. CONCLUSIONS

Aquesta eina es basa en un entorn controlat per consola capaç d'executar ordres per facilitar l'enumeració i documentació dels diferents objectius en un pentest. La principal característica és que, amb els coneixements apropiats, es poden ampliar i personalitzar les diferents funcionalitats de les que es disposa. Quan es vol consultar la informació recollida

es pot fer mitjançant el portal web que s'ha preparat. Aquesta web mostra de manera ràpida i simple el contingut de la base de dades dedicada a aquesta eina. A més també es pot activar un listener capaç d'executar ordres a distància i per veu.

Pel que fa als resultats obtinguts dels objectius inicials:

- Base de Dades: S'ha aconseguit construir una estructura capaç de representar i desar les dades obtingudes de manera coherent amb la informació que representen. A més permet realitzar canvis i adaptar-se amb poc esforç a la nova estructura.

- Entorn: Es pot afirmar que s'ha aconseguit desenvolupar un entorn altament modificable i personalitzable requerint únicament coneixements suficients com per programar noves funcionalitats o modificar les existents. Al estar programat de manera modulada és altament resistent i adaptable a nous canvis.

- Web: Es disposa d'una web simple que realitza consultes de manera segura a la nostra base de dades i que mostra de manera eficaç la informació proporcionada. Actualment cal treballar la seva apariencia per tal de millorar i facilitar l'experiència de l'usuari, però la funcionalitat es considera acabada.

- Integració amb Alexa: S'ha desenvolupat un servidor semblant al de la web capaç de rebre ordres pel protocol HTTP i executar-les. Aquest servidor funciona a la perfecció i s'hi pot accedir des de qualsevol navegador. Pel que fa a les Skills programades també s'ha comprovat la seva funcionalitat. Actualment l'usuari és capaç d'executar ordres per veu des de qualsevol part del mon es disposi d'una Alexa a la qual se li ha afegit la nostra skill. Cal recordar que l'usuari sabrà si l'ordre s'ha executat correctament o no, però no el seu resultat.

Cal tenir present que falta acabar d'ajustar i perfeccionar alguns dels objectius proposats per tal d'aconseguir la màxima funcionalitat desitjada, però actualment disposem de resultats altament satisfactoris. Amb tota la feina realitzada es pot afirmar que s'ha aconseguit desenvolupar una éina amb una funcionalitat simple, però amb un gran potencial per als usuaris dedicats al món de la ciberseguretat ofensiva.

Com a treball futur cal seguir augmentant les funcionalitats creant nous scripts per tal de potenciar i millorar el poder d'aquesta eina. Si es creu necessari també es pot treballar en la interfície de la web per millorar l'experiència de l'usuari final. En conclusió, ara simplement cal millorar el sistema.Actualment el seu estat es considera acabat i preparat per créixer amb infinites possibilitats.

9. AGRAIMENTS

Personalment vull agrair i dedicar aquest treball a tot l'equip de resposta d'incidents de ciberseguretat (CIRT) de Valeo, no només per l'ajuda i suport durant aquest projecte, sinó també per tots els ànims, ganes i facilitats proporcionades que des del primer dia van fer que em sentís un membre més d'aquest gran equip de professionals.

També vull donar les gràcies a en Ramon, el meu actual tutor del treball, i agrair-li la plena dedicació i preocupació perquè pogués realitzar aquesta feina. Des de tota la documentació proporcionada a l'inici del projecte fins la paciencia que ha demostrat tenir en cada una de les correccions dels informes entregats.

El resultat d'aquest treball queda a plena disposició de tot aquell que en vulgui fer ús

10. FONTS D'INFORMACIÓ I BIBLIOGRAFIA

[1]. What is Penetration Testing
https://www.imperva.com/learn/application-security/penetration-testing/

[2]. Red Team vs Blue Team
https://securitytrails.com/blog/cybersecurity-red-blue-team

[3]. What is Alexa
https://developer.amazon.com/es-ES/alexa

[4]. Tutorial Alexa Skill with Python:
https://developer.amazon.com/en-US/alexa/alexa-skills-kit/resources/training-resources/alexa-skill-python-tutorial

[5]. What is Scrum?
https://en.wikipedia.org/wiki/Scrum_(software_development)

[6]. Python - PostgreSQL wiki
https://wiki.postgresql.org/wiki/Python

[7]. HOWTO de programación de sockets
https://wiki.python.org/moin/HowTo/Sockets

[8]. Requests: HTTP para Humanos
https://requests.readthedocs.io/es/latest/